НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

____ _____ Теплоенергетичний факультет _____

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2019р.

ДИПЛОМНА РОБОТА

на здобуття ступеня бакалавра

з напряму підготовки 6.050103 “ Програмна інженерія “

на тему: Web-система трекінгу та ідентифікації об’єктів у відеопотоці

Виконав: студент 4 курсу, групи ТІ-51

_____Міщенко Олександра Ярославівна_____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник к.т.н. Шалденко Олексій Вікторович_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший , бакалаврський

Напрям підготовки 6.050103 “ Програмна інженерія”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)
”___”_____2019р.

ЗАВДАННЯ
на дипломну роботу студенту
Міщенко Олександрі Ярославівні
(прізвище, ім’я, по батькові)

1.Тема роботи Web-система трекінгу та ідентифікації об’єктів у відеопотоці
Керівник роботи к.т.н. Шалденко Олексій Вікторович
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ” ” 201 р. № _____

2. Строк подання студентом роботи _____
3. Вихідні дані до роботи: мова програмування Python.
4.Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити) Розробити систему трекінгу та ідентифікації об'єктів у відеопотоці. Розробити програмний інтерфейс системи.

5. Перелік ілюстративного матеріалу: схеми архітектури додатку, діаграма прецедентів системи, зразки розробленого інтерфейсу додатку.
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ” ”_____201 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Вивчення та аналіз задачі		
2	Розробка архітектури та загальної структури системи		
3.	Розробка структур окремих підсистем		
4.	Програмна реалізація системи		
5.	Оформлення пояснювальної записки		
6.	1 Захист програмного продукту		
7.	2 Передзахист		
8.	Захист		

Студент _____ Міщенко О.Я.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Шалденко О.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

бакалаврської дипломної роботи Міщенко Олександри Ярославівни тему «Web-система трекінгу та ідентифікації об’єктів у відеопотоці»

Мета роботи – дослідження алгоритмів трекінгу та ідентифікації об’єктів у відеопотоці та створення веб-застосунку, який за допомогою цих алгоритмів дозволяє обробляти дані за камер відеоспостереження та зберігати інформацію про місце та час знаходження людини на відео.

У процесі роботи досліджувалися програмні продукти та технології, які застосовуються для трекінгу та ідентифікації об’єктів.

Результатом роботи є працююча та протестована веб-система трекінгу та ідентифікації об’єктів, опис процесу побудови системи та рекомендації та інструкції щодо використання системи.

Розробка системи здійснювалась в середовищі Visual Studio Code.

Обсяг роботи 61 сторінка, 21 ілюстрацію, 2 формули, 17 використаних джерел, 5 додатки.

Ключові слова: трекінг об’єктів, знаходження об’єктів, ідентифікація людини, обробка зображення, нейронні мережі, комп’ютерний зір

ABSTRACT

Bachelor diploma work by Mishchenko Oleksandra "Web tracking and identification system on video stream"

The purpose of the work is to study object tracking and identification algorithms and to create a web-application which processes data using these algorithms and saves the data about a person, its location and time of appearance on video.

In the framework of this work software products and technologies for object tracking and identification tasks were studied.

The result of the work is a running and tested system for object tracking and identification, a description of the process of building the system and recommendations and instructions for using the system.

System development was performed in Visual Studio Code environment.

Explanatory note: 61 p., 21 fig., 2 formulas, 17 references.

Keywords: object tracking, object detection, person identification, computer vision, neural networks, opencv

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД	–	база даних
Трекінг	–	відстеження
SDK	–	бібліотека
IDE	–	інтегрована середа розробки

ВСТУП

Останнє десятиріччя демонструє стрімкий ріст обсягів цифрової інформації та поширення систем, які її зчитують, обробляють та використовують. Серед загального обсягу інформації велика частка візуальної інформації збирається через системи відеоспостереження. Магазини, зупинки, громадський транспорт, житлові будинки, вулиці забезпечені сучасними системами відеоспостереження, завдяки яким можна зробити життя людини безпечнішим та простішим: ідентифікувати небезпечних злочинців, знаходити зниклих людей, запобігати терактам.

Водночас, такі системи використовуються для автоматизації процесів як у побутовій, так і бізнес-сфері. Компанії використовують алгоритми розпізнавання, відстеження та ідентифікації для контролю працівників або для аналізу поведінки людей з метою покращення якості сервісу.

Наприклад, у випадку супермаркетів та магазинів можна відстежувати навантаження на магазини, взаємодію між покупцями та продавцями, активність самих продавців та попереджувати крадіжки. Найкращим прикладом актуальності цієї задачі є Китай, який використовує такі системи для ідентифікації злочинців, ідентифікації співробітників в аеропортах або відстеження пересування людей у “безпечних зонах”[1]. Якщо людина покидає такі зони, поліція має право зупинити людину та допитати її.

Для офісних компаній можливим є відстеження часу, який працівник проводить в офісі та прилеглих приміщеннях, скільки часі проводять в ньому люди, які не є працівниками або повідомляти, якщо стороння людина знаходиться у приміщенні, до якого в неї немає доступу. У випадку пожежі ці дані можуть допомогти швидше знайти та врятувати людину.

Технології, які при цьому використовуються, розвиваються дуже швидко, адже області їх застосування не обмежуються вище наведеними прикладами – практика використання технологій розпізнавання перманентно розширюється. Особливо перспективними у задачі розпізнавання образів та знаходження об'єктів є

згорткові нейронні мережі, які за допомогою згортки дозволяють ефективно знаходити ознаки, а завдяки багатошаровій архітектурі знаходити комплексні особливості серед великих обсягів даних.

Тим не менш, класичні алгоритми також широко використовуються завдяки своїй простоті у реалізації та швидкості у роботі.

Метою дипломної роботи є побудова системи ідентифікації та трекінгу об’єктів у відеопотоці на основі вже існуючих актуальних алгоритмів. Отримані результати зберігаються у базу даних для можливості подальшого аналізу та використання.

Головними задачами дослідження є:

1. Вивчення й аналіз існуючих систем, що дозволяють розпізнавати об’єкти.

2. Вивчення й аналіз актуальних алгоритмів для відстеження та розпізнавання об’єктів.

3. Порівняльний аналіз конкурентних систем та практик.

4. Розробка власного варіанту системи ідентифікації та трекінгу об’єктів на базі найкращих практик, підходів та інструментів.

Результати роботи пропонуються до використання компаніям, які займаються технічною підтримкою з метою відстеження знаходження співробітника на своєму робочому місці.

Впровадження запропонованої системи призведе до підвищення ефективності роботи компанії.

Для розробки програмного додатку було обрано мову програмування Python та фреймворк Django для створення веб-додатків. Розробка відбувалась у середовищі програмування Microsoft Visual Studio Code, візуальний інтерфейс було створено засобами HTML, CSS та JavaScript. В якості системи управління базами даних було використано MongoDB.

1. ПОСТАНОВКА ЗАДАЧІ З ТРЕКІНГУ ТА ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ У ВІДЕОПОТОЦІ

Проблема автоматизації та оптимізації процесів як загальне прагнення отримати найкращий результат за найменшу кількість часу із витратою найменших зусиль є спільним для будь-якого технологічного процесу. Сучасний рівень розвитку технологій надає велику кількість простих та комплексних інструментів, серед яких в останнє десятиріччя широкої популярності набули наука про дані та машинне навчання.

Особливої уваги заслуговують проблеми, які можуть бути вирішені за допомогою даних з відеопотоку, адже кожного дня людина контактує з великою кількістю інформаційних систем.

Для окремих сфер діяльності можливість контролю та спостереження за працівниками, клієнтами та людьми без потреби цілодобового очного нагляду є дуже затребуваною. Наприклад, трекінг консультантів та продавців та їх взаємодії із покупцями дозволяє визначити працівників, які демонструють низьку вмотивованість у роботі, що призводить до її низької результативності та у кінцевому рахунку – до втрати прибутку. Трекінг покупців у супермаркетах дозволяє знайти найбільш популярні та «мертві» зони або викрити крадіїв, а відстеження та ідентифікація людей в аеропортах та публічних місцях скупчення дає змогу попередити злочини, теракти або знайти розшукуваних зловмисників.

Незалежно від розміру та обсягів капіталу, компанії прагнуть впроваджувати технології машинного навчання для аналізу даних та автоматизації процесів. Одним із пріоритетних шляхів підвищення якості роботи та ефективності працівників компаній стає впровадження систем трекінгу та ідентифікації для вирішення цієї задачі. Статистика часу, проведеного працівником у робочій кімнаті та кімнаті відпочинку дозволяє обґрунтовано обраховувати показники того,

наскільки точною є оцінка часу для виконання задач, а також для створення ключових показників ефективності. Крім того, такі дані можуть допомогти при вирішенні конфліктних ситуацій, які стосуються звільнення працівників.

Дослідження таких алгоритмів широко розповсюджується, а області застосування майже необмежені. Тому було запропоновано дослідити їх можливості на прикладі Web-системи трекінгу та ідентифікації для менеджерів, яка надає аналітику щодо присутності та поведінки працівників на роботі.

Як результат дослідження пропонується створення програмного застосунку, функціями якого є:

- забезпечення користувача (менеджера) персональним кабінетом, в якому можна додавати власні камери спостереження або завантажувати відео з камер та отримати статистику до кожного закріпленого за користувачем працівника;
- визначення та відстеження людини у відеопотоці:
- встановлення особи кожної наявної в кадрі людини;
- забезпечення користувача зручним графічним інтерфейсом.

Для цього необхідно:

- провести дослідження сучасних методів знаходження, трекінгу та ідентифікації людини;
- провести порівняльний аналіз існуючих інструментів;
- обрати набор бібліотек для роботи з відео та обробки зображень;
- створити зручний веб-додаток з визначеним вище функціоналом для користувача.

2. ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ

Так як система трекінгу об’єктів є дуже затребуваною у сфері автономних автомобілів та у системах безпеки, програмні засоби, які сконцентровані на відстеженні людей є досить поширеними.

Прикладами таких сервісів SOLOSHOT3[2], Vision4ce[3] та ART[4].

2.1. SOLOSHOT3

Продукт SOLOSHOT3 – камера, яка дозволяє записувати відео у високій якості та має вбудовані алгоритми відстеження об’єктів. Однією з її переваг є якісний зум (збільшення зображення у 65 разів не втрачаючи якості) та автоматичне відстежування заданого об’єкту у кадрі.

Для того, щоб отримати дані з камери, використовується зручний інтерфейс, написаний на мові програмування Python. Проте, цілісної системи для роботи з даними з камер та відображення їх немає. Крім того, виробник не рекомендує використовувати цю камеру для зйомок в приміщеннях та місцях великого скупчення людей, що є суттєвим обмеженням в можливих сферах використання.

2.2. Vision4ce

Компанія Vision4ce пропонує свою камеру разом зі своїм програмним забезпеченням для задач трекінгу об’єктів. Їх програмний продукт працює з операційною системою Windows або Linux та дозволяє розпізнавати одночасно багато об’єктів на кадрі та відстежувати їх місцезнаходження, а також класифікувати об’єкти на рухомі та нерухомі.

На рисунку 2.1 зображено приклад роботи їх програми.

Рисунок 2.1 – Приклад розпізнавання об’єкту за допомогою Vision4ce.

2.3. ART

Компанія ART, в свою чергу, пропонує одразу три системи: SMARTTRACK, TRACKPACK та ARTTRACK.

Система SMARTTRACK оптимізована для мобільних пристроїв та має помірну ціну. Вона здатна знаходити об’єкти на кадрі та відстежувати їх місцезнаходження.

Система TRACKPACK розширює функціонал SMARTTRACK тим, що може знаходити невеликі об’єкти та класифікувати чи є об’єкт рухомим чи нерухомим. Рекомендована для розпізнавання обличь та жестів.

Система ARTTRACK, як найбільш розвинена, має кращу якість розпізнавання та є стійкою до швидких рухомих об’єктів та поганих світлових умов на кадрі.

2.4. Висновки до розділу

Таким чином було розглянуто три основних доступних системи для ідентифікації та трекінгу об’єктів у відеопотоці. Існують також і різні, більш складні системи, такі як в Китаї, проте вони є закритими та не для використання в широкому доступі.

В той же час, досліджені системи не розв’язують задачу повністю. Вони сконцентровані лише на відстеженні об’єктів у кадрі. Жодна з систем не має функціоналу ідентифікації та зручного інтерфейсу для використання.

3. АНАЛІЗ МОЖЛИВОСТЕЙ АЛГОРИТМІВ ТРЕКІНГУ ТА ІДЕНТИФІКАЦІЇ

Області комп’ютерного зору та машинного навчання розвиваються дуже стрімко. Згідно з CVPR[5], однієї з найбільших конференцій с комп’ютерного зору, кількість поданих дослідницьких статей зросла на 56% у порівнянні з минулим роком. Особливо простежується інтерес до нейронних мереж, які за останні кілька років складають від 20% до 25% поданих статей на основні конференції з комп’ютерного зору та глибокого навчання [6]. Подивитись тенденцію можна на рисунку 3.1.

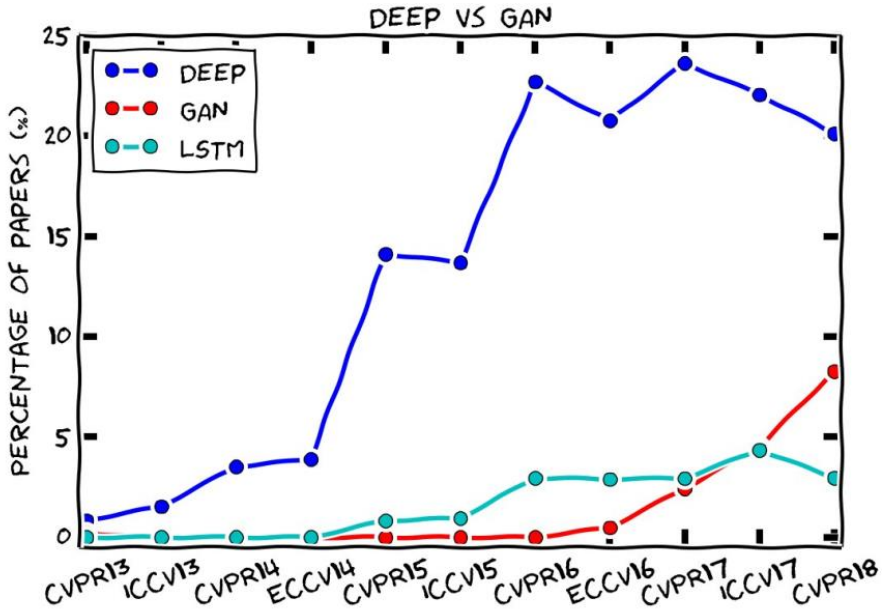

Рисунок 3.1 – Графік процентного співвідношення статей на тему глибокого навчання на різних конференціях в різні роки [6]

Задачі знаходження, розпізнавання та ідентифікації є одними з основних задач комп’ютерного зору. До того ж, часто вони є взаємопов’язаними. Наприклад, для трекінгу об’єкту необхідно задати початкове місцезнаходження об’єкту, а, отже, необхідний точний алгоритм для знаходження такого типу об’єктів. Якість

ідентифікації об’єктів залежить від якості поданого зображення об’єкту, що також впирається в можливості алгоритмів відстеження та розпізнавання.

3.1. Аналіз існуючих алгоритмів для знаходження об’єктів

Найбільш простим алгоритмом для знаходження об’єктів є метод Віоли-Джонса. Незважаючи на те, що він був розроблений у 2001 році, він все ще широко використовується для пошуку об’єктів на зображеннях в реальному часі. Він швидко працює та не потребує великих обчислювальних ресурсів – більшість камер за замовчуванням використовують саме цей детектор. Проте, великим його недоліком є відсутність стійкості до різних поз або зовнішніх умов.

Іншим класичним алгоритмом знаходження об’єктів є Histogram of Oriented Gradients Detector (HOG). Найбільшою його перевагою є те, що, відносно методу Віоли-Джонса він є стійким до поз, тобто можливість знаходження об’єкту не залежить від кута повороту об’єкту. Серед недоліків необхідно виділити те, що цей метод не може знаходити маленькі об’єкти, не є стійким до світлових умов та якість розпізнавання гірша за рахунок того, що бокси знайденого об’єкту захоплюють область навколо об’єкту.

Одні з найкращих результатів в області розпізнавання осіб досягається за допомогою використання згорткових нейронних мереж. Успіх обумовлений можливістю обліку двовимірної топології зображення, на відміну від багатошарового перцептрона. Сучасні state-of-art алгоритми дійсно показують кращу продуктивність та більшу точність, проте кожен з них має свої недоліки. Розглянемо три основних популярних алгоритми, а саме Faster R-CNN, YOLO та SSD.

Алгоритм Faster R-CNN є найбільш точним серед відомих детекторів та є стійким до перетворень: поворотів об’єкту, різноманітних ракурсів та розмірів.

Проте він не підходить для роботи в реальному часі, бо опрацювання кадру займає забагато часу (близько 47 секунд на кожне зображення).

Наступний алгоритм, YOLO або You Only Look Once працює настільки швидко, що його можна використовувати для роботи в реальному часі, проте він показує погану якість в розпізнаванні малих, дуже великих або перевернутих об’єктів.

Алгоритм SSD або Single Shot Detection вирішує проблему YOLO щодо великих та перевернутих об’єктів, проте проблема низької якості знаходження малих об’єктів залишається.

Так як система потребувала невеликих обчислювальних ресурсів та швидкої обробки відео, в якості робочого алгоритму було обрано метод Віоли-Джонса. Розглянемо його більш детально.

3.1.1. Метод Віоли-Джонса

Метод Віоли-Джонса має 4 основні етапи:

1. Пошук ознак Гаара.

Ознака - відображення $f: X = D_f$, де D_f- множина допустимих значень ознаки. Якщо задані ознаки $f_1 \dots f_n$, то вектор ознак $x = (f_1(x), \dots, f_n(x))$ називають описом ознаки об’єкту $x \in X$. При цьому множина $X = D_{f1} * \dots * D_{fn}$ називають простіром ознак.[6]

Ознаки поділяються на наступні типи в залежності від множини D_f:

- бінарна ознака: $D_f = \{0,1\}$;
- номінальная ознака: D_f є кінцевою множиною;
- порядкова ознака: D_f є кінцевою впорядкованою множиною;
- кількісна ознака: D_f є множиною дійсних чисел.

Ознаки Гаара описують риси об’єкту, який необхідно знайти.

На рисунку 3.2 зображено ознаки Гаара, які використовуються в методі Віоли-Джонса.

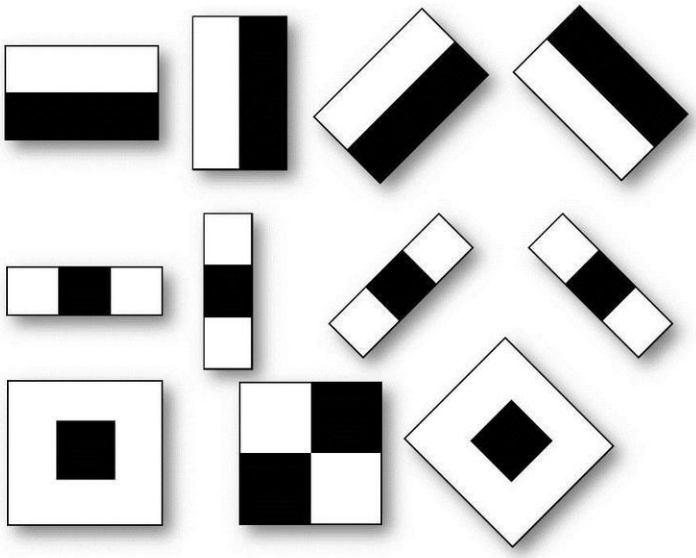

Рисунок.3.2 – Прямокутні ознаки Гаара [7]

Кожна з ознак обчислюється таким чином: від суми значень пікселів, які закриває світла частина ознаки віднімається відповідна сума значень пікселів, які закриває темна частина ознаки.

2. Перетворення зображення в інтегральний формат

Для того, щоб прискорити обчислення ознак Гаара, було вирішено звести зображення до інтегрального формату.

Інтегральне представлення зображення - це матриця, розмір якої співпадає з розміром вихідного зображення. Кожен елемент цієї матриці дорівнює сумі значень пікселів лівіше та вище до відповідного пікселя відносно елементу матриці. На рисунку 3.3 зображено приклад побудови інтегрального представлення зображення.

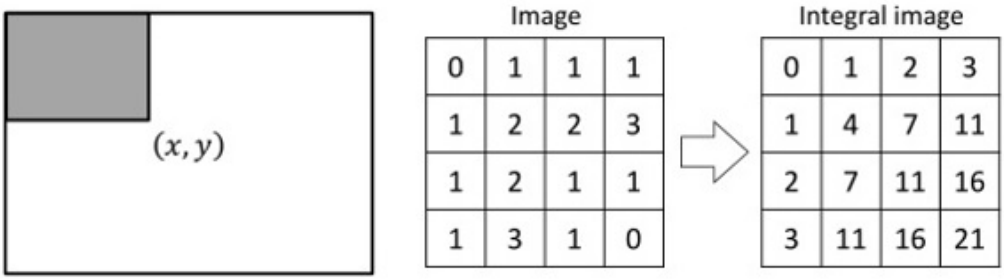

Рисунок 3.3 – Інтегральне представлення зображення [8]

3. Навчання алгоритму за допомогою алгоритму Adaboost.

Так як з кожної ознаки Гаара обчислюється велика кількість ознак по всьому зображенні, необхідно їх фільтрувати. Наприклад, на рисунку 3.4 зображено застосування двох таких ознак. І якщо для очей надана ознака має сенс, так як області біля очей часто темніші за область носу, то та сама ознака застосована на щоці не матиме сенсу.

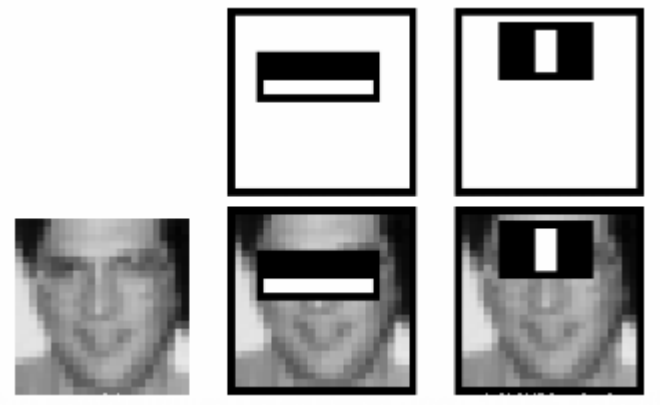

Рисунок 3.4 – Застосування ознак Гаара[8]

Саме тому необхідно серед усіх можливих сформованих ознак обрати найкращі. Це досягається за допомогою алгоритму Adaboost, який одночасно відбирає ознаки та навчає класифікатор їх використовувати. Цей алгоритм будує “сильний” класифікатор як лінійну комбінацію “слабких” класифікаторів шляхом створення каскаду.

4. Побудова каскаду класифікаторів

На рисунку 3.5 представлено каскадну архітектуру.

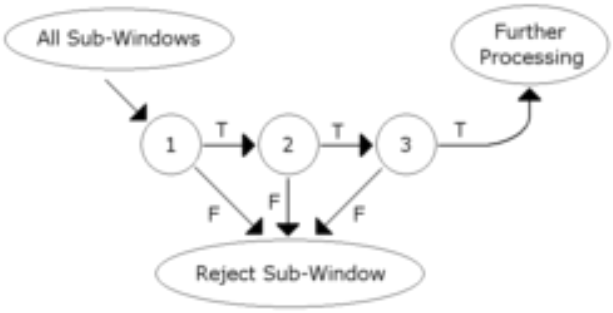

Рисунок 3.5 — Каскадна архітектура

Для того, щоб обійти зображення повністю та мати можливість розпізнавати об’єкти різних масштабів, застосовується принцип вікон, тобто вікно фіксованого розміру проходить по всьому зображенню. Формуються каскади, кожен з яких містить окрему комбінацію «слабких» класифікаторів та групу ознак, яку необхідно знайти. Якщо поточний каскад не знаходить задані ознаки та приймає рішення, що потрібного об’єкту вікно не має – наступний каскад не застосовується до вікна, вікно відхиляється, а алгоритм переходить до наступного вікна.

3.2. Аналіз існуючих алгоритмів трекінгу об'єктів

Алгоритми знаходження об’єктів знаходять кожен об’єкт одноразово, тому щоб знайти цей самий об’єкт на наборі кадрів необхідно кожного разу застосовувати цей алгоритм. Це не є ефективним з точки зору обчислень та не гарантує, що об’єкт буде знайдено на кожному за кадрів, так як не всі детектори є стійкими до різних поз та ракурсів.

Ця проблема вирішується за допомогою алгоритмів трекінгу об’єктів, які дозволяють відстежувати об’єкт на наборі кадрів. Спочатку використовуються алгоритм знаходження об’єкту для того, щоб ініціалізувати алгоритм трекінгу. Далі, по координатам об’єкту з попереднього кадру робиться обчислення координат об’єкту в наступному кадрі.

Бібліотека OpenCV містить реалізацію 8 основних алгоритмів для відстеження об’єктів: BOOSTING, MIL, TLD, MEDIANFLOW, MOSSE, Predator та GOTURN. Відповідно до тестів, зроблених ресурсом learnopencv.com[9], краще всього використовувати Predator або GOTURN.

3.2.1. Трекер GOTURN

Трекер GOTURN - єдиний з запропонованих алгоритмів, який працює на нейронних мережах, тому він має вищу точність ніж у конкурентів, а проста архітектура забезпечує високу швидкість виконання алгоритму.

На рисунку зображено принцип роботу алгоритму GOTURN.

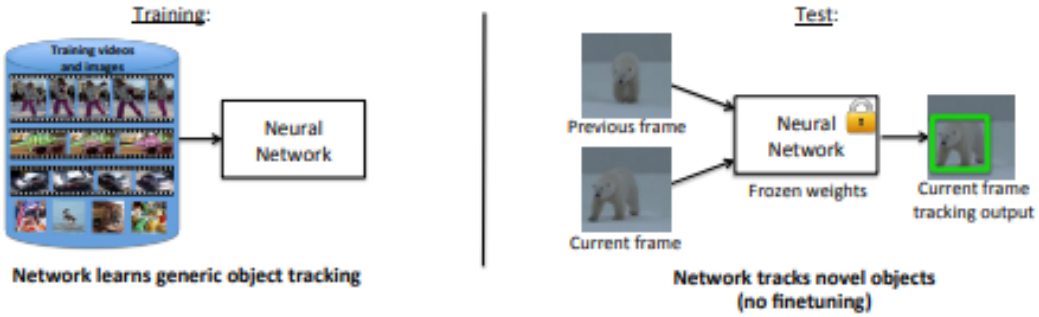

Рисунок 3.6 – Принцип роботи алгоритму GOTURN

На першому кадрі (або на попередньому) місцезнаходження та розміри об’єкту відомо, та з кадру вирізається об’єкт з двічі більшими параметрами для того щоб залучити також область біля об’єкту. Об’єкт завжди централізований в вирізаній області зображення.

Задача знайти місцезнаходження об’єкту на наступному кадрі. Об’єкт також вирізається за відомими з попереднього кадру параметрами, проте не централізується.

Ці дані подаються на вхід згортковій нейронній мережі, яка навчається за попереднім кадром передбачати місцезнаходження об’єкту на наступному кадрі. Схематична робота нейронної мережі наведена на рисунку 3.6.

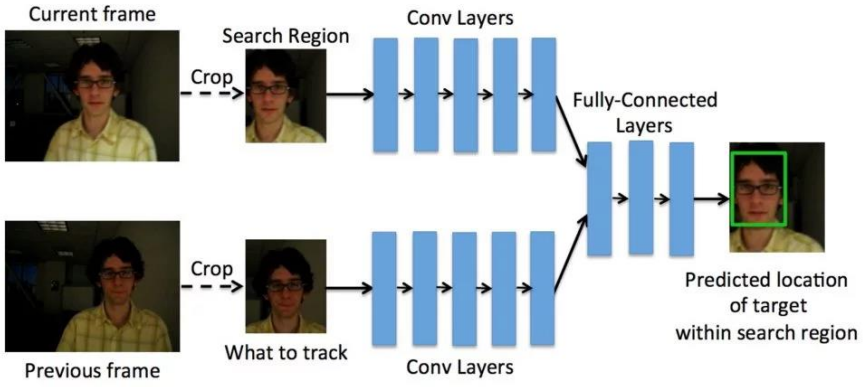

Рисунок 3.7 - Схематична робота нейронної мережі, яка лежить в основі GOTURN алгоритму

Архітектура нейронної мережі представляє собою 5 згорткових шари (перші п’ять шарів архітектури CaffeNet) та три повнозв’язних шари, останній з яких повертає нейрони: верхня та нижня координати боксу шуканого об’єкту.

Модель написана та натренерована за допомогою фреймворку глибокого навчання Caffe.

3.3. Аналіз існуючих алгоритмів для ідентифікації об'єктів

Для ідентифікації об’єктів широко використовуються сіамські нейроні мережі. Найбільша перевага сіамських нейронних мереж полягає у тому, що вони дозволяють розрізняти об'єкти одного класу по єдиному зображенню (наприклад, для ідентифікації окремої людини в натовпі достатньо завантажити одне фото цієї людини). На вхід подаються два зображення. Алгоритм кодує ці зображення та знаходить відстань між ними. Чим менше відстань – тим більша вірогідність того, що на фото один і той самий об’єкт.

Альтернативою згортковим нейронним мережам для ідентифікації об’єктів на зображенні є класичні алгоритми галузі комп'ютерного зору (наприклад, SIFT), проте вони дуже програють нейронним мережам у точності.

Тому, для ідентифікації людей в дипломній роботі було використано алгоритм FaceNet, який є реалізацією сіамських нейронних мереж.

3.3.1. Алгоритм FaceNet

FaceNet - сіамська нейронна мережа, яка навчається перетворювати зображення облич у евклідовий простір, де відстань відповідає мірі схожості обличь.

Сіамські нейронні мережі були запропоновані на початку 1990х років Бромлі та ЛеКуном для вирішення задачі верифікації підпису як проблеми подібності зображень [10]. Сіамська нейронна мережа складається із мереж близнюків, які приймають різні вхідні дані, але об'єднані однією функцією втрат, при цьому мережі близнюки мають абсолютно однакові ваги.

На рисунку 3.7 показана принцип роботи сіамських мереж.

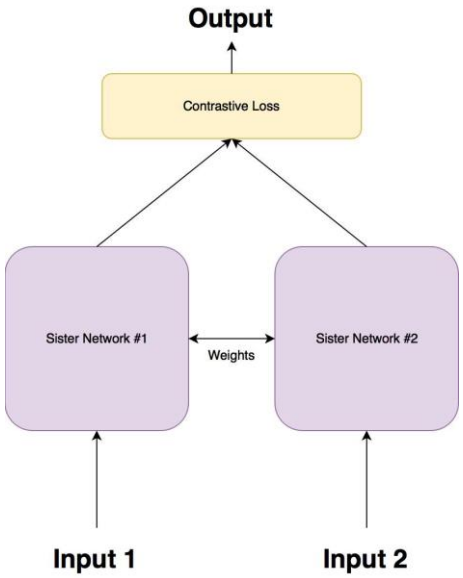

Рисунок 3.8 - Архітектура сіамських нейронних мереж.

Існує декілька функцій втрат, які використовуються для роботи з сіамськими нейронними мережами: “контрастна” функція втрат (contrastive loss) та “функція триплетів” (triplet loss).

Для contrastive loss використовують пари зображень та розраховують її за формулою 1:

$$L_{contrastive}(x_0, x_1, y) = \frac{1}{2} y \left|\left|f(x_0) - f(x_1)\right|\right|_2^2 + \frac{1}{2}(1-y)\{max\left(0, m - \left|\left|f(x_0) - f(x_1)\right|\right|_2^2\right)\}^2 [1],$$

де x_0 та x_1- вхідні зображення, y – змінна, яка приймає значення 0 або1 в залежності від того, чи знаходиться на обох зображеннях одна й та сама людина, f - досліджувана функція, яка перетворює зображення у вектор в евклідовому просторі.

Алгоритм FaceNet використовує triplet loss, так як замість двох ідентичних нейронних мереж його архітектура містить три нейронні мережі. На рисунку 3.7 зображено схему роботи FaceNet.

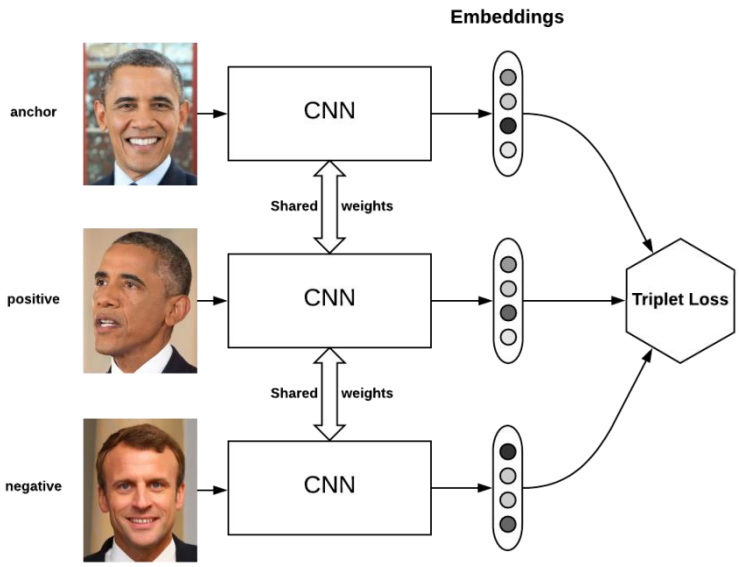

Рисунок 3.9 – Схема роботи FaceNet

На вхід FaceNet подаються три зображення, два зображення належить одній й той самій особі (вони позначаються anchor та positive), третє зображення – сторонній особі (negative). Задача алгоритму мінімізувати відстань між однаковими особами та максимізувати відстань між різними особами. Це досягається

мінімізацією функції втрат – функції триплетів, яка розраховується за формулою 2:

$L_{triplet} = \left|\left|f^a - f^p\right|\right|_2^2 - \left|\left|f^a - f^n\right|\right|_2^2 + \propto$ [2],

де a, p та n – три вхідні зображення, $\propto$ – , f - досліджувана функція, яка перетворює зображення у вектор в евклідовому просторі.

3.4. Висновки до розділу

Таким чином у даному розділі було розглянуто основні алгоритми, які використовуються для трекінгу, знаходження та ідентифікації об’єктів, обґрунтовано вибір алгоритмів для кожної з задач та розглянуто їх більш детально.

4. ЗАСОБИ РОЗРОБКИ

Для виконання поставленої задачі було використано бібліотеки OpenCV та Dlib.

4.1. Бібліотека OpenCV

Бібліотека OpenCV (Open Source Computer Vision Library) – це бібліотека з відкритим вихідним кодом, яка містить близько 2500 алгоритмів комп'ютерного зору, обробки зображень та чисельних алгоритмів. Вона реалізована на мові C++, проте має інтерфейси на мовах програмування Python, Java, Ruby, Matlab.

В рамках даної роботи було обрано інтерфейс OpenCV-Python, так як він поєднує в собі швидкість виконання оригінального C++ коду та простоту написання коду мовою програмування Python. До того ж, підтримка Numpy – високооптимізованої бібліотеки для чисельних обчислень, розширяє можливості використання бібліотеки OpenCV – спрощує математичні обчислення та дає змогу легко відображати результати за допомогою бібліотек Matplotlib та SciPy.

Бібліотека OpenCV складається з 12 основних невеликих модулів, розділених по їх функціональному використанню:

- opencv_core – базові структури, обчислення;
- opencv_imgproc – обробка зображень;
- opencv_highgui – завантаження/зберігання відео;
- opencv_ml – методи та моделі машинного навчання;
- opencv_features2d – дескриптори;
- opencv_video – аналіз руху та відстеження об'єктів;
- opencv_objdetect – знаходження об'єктів на зображенні;

- opencv_calib3d – робота з камерою;
- модулі для невикористаного коду, старого коду для підтримки сумісності старих реалізацій та пришвидшення алгоритмів за рахунок виконання частини обчислень на відеокартах.

4.2. Знаходження об’єктів за допомогою OpenCV

Як було зазначено в пункті 3.2, в даній роботі для знаходження обєктів використовувався метод Віоли-Джонса, в бібліотеці OpenCV відомий як Cascade Classifier.

Для того, щоб користуватися цим детектором, необхідно спочатку завантажити та вказати шлях до XML файлів претренерованих класифікаторів.

Приклад визначення обличчя представлений на рисунку 4.1.

Рисунок 4.1 – Демонстрація визначення обличчя на відео.

Координати знайденого обличчя далі передається до алгоритму відстеження об’єктів, також реалізованому за допомогою OpenCV.

Для того, щоб оновлювати інформацію щодо появи нових людей в кадрі, Cascade Classifier виконується через кожні 30 кадрів (близько однієї секунди).

4.3. Трекінг об’єктів за допомогою OpenCV

Як було зазначено в пункті 3.4, в даній роботі для трекінгу обєктів використовувався алгоритм GOTURN, який є реалізований в бібліотеці OpenCV.

Для того, щоб відстежувати одночасно декілька об’єктів на кадрі, необхідно спочатку створити об’єкт класу cv2.MultiTracker. Потім, для кожного знайденого об’єкту створюється окремий об’єкт — трекер GOTURN та ініціалізується з відповідними параметрами об’єкту, а клас cv2.MultiTracker об’єднує ці окремі об’єкти в єдиний ансамбль.

Так як алгоритми відстеження об’єктів не оновлюються автоматично з появою нових об’єктів в кадрі, необхідно їх комбінувати з алгоритмами знаходження об’єктів. Кожен раз, коли кількість знайдених об’єктів детектором не співпадає з кількістю об’єктів, знайдених трекером, необхідно:

1. Додати нові трекери до ансамблю, якщо об’єктів стало більше;
2. Порівняти попередні координати з межами кадру. Якщо об’єкт знаходився у межах кадру та неможливо уявити ситуацію, при якій він за даний проміжок часу може вийти за поле зору, тоді відновити трекер цього об’єкту;

4.4. Бібліотека Dlib

Бібліотека Dlib – це бібліотека з відкритим вихідним кодом, написана на мові програмування C++, яка містить в собі алгоритми машинного навчання та інструменти для вирішення реальних проблем. ЇЇ використовують у сферах робототехніки, мобільних телефонів, програмуванні пристроїв тощо. Має інтерфейс на мові програмування Python.

Головними її особливостями є:

- Документація. Бібліотека має повну та зрозумілу документацією з поясненням роботи алгоритмів та прикладами використання цих алгоритмів на мовах програмування C++ та Python.
- Висока якість коду. Бібліотека Dlib працює на трьох основних платформах: Windows, Linux та Mac OS X. Також, кожен програмний модуль покритий тестами, що знижує ризик потрапляння непрацюючого коду до бібліотеки.
- Реалізовані алгоритми машинного та глибокого навчання, такі як метод опорних векторів та многошаровий перцептрон.
- Реалізовані чисельні алгоритми лінійної алгебри.
- Засоби для обробки зображень. Бібліотека дозволяє не тільки зчитувати та обробляти зображення, а й знаходити об’єкти, виділяти ознаки чи ідентифікувати людей.
- Засоби для роботи з мережами.
- Інструменти для створення графічних інтерфейсів.
- Наявність загальних інструментів для роботи з файлами, терміналом та роботою з пам’яттю.

4.5. Ідентифікація об’єктів за допомогою бібліотеки Dlib

Як було зазначено в пункті 3.6, ідентифікація об’єктів здійснюється за допомогою сіамської нейронної мережі FaceNet.

Бібліотека Dlib містить модуль face_recognition, в якому реалізовані алгоритми для розпізнавання облич на зображенні та встановлення їх особистості при наявності заданих зображень, які вже повинні бути представленні у вигляді векторів.

Цей модуль дозволяє отримати векторну репрезентацію зображення, яке потім порівнюється з векторами зображень з бази даних працівників. Для цього вираховується евклідова відстань між заданим зображенням та кожним записом з бази даних, серед результатів обчислень обирається найменша відстань і якщо ця відстань менше граничного значення (0.5, цей параметр може змінюватись), то вважається, що на фото зображена одна й та сама особа.

4.6. Висновки до розділу

У даному розділі було розглянуто основні засоби розробки для реалізації алгоритмів трекінгу, знаходження та ідентифікації об’єктів та розглянуто принципи користування даними інструментами для вирішення поставлених.

5. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ АВТОМАТИЗОВАНОГО РОЗГОРТАННЯ СИСТЕМ

Для побудови веб системи було використано веб фреймворк Django 2.0, так як він добре інтегрується з сервісами для машинного навчання та використовує мову програмування Python.

5.1. Фреймворк Django

Фреймворк Django - це високорівневий веб-фреймворк, який реалізований на основі архітектури MVC (Model-View-Controller). Часто цю архітектуру називають MVT (Model-View-Template).

Особливості Django:

- будь-який запит оброблюється програмно та перенаправляється на свою адресу (url);
- розділ контенту та представлення за допомогою шаблонів;
- абстрагування від низького рівня баз даних.

Застосунок Django складається з чотирьох основних компонентів:

1. Модель даних. Модель – важлива частина застосунку, яка постійно звертається до даних при будь якому запиті. Модель є стандартним Python-класом. Об’єктно-орієнтований маппер (ORM) забезпечує таким класам доступ безпосередньо до баз даних, що дає можливість не писати запроси безпосередньо на SQL.

2. Представлення. В моделі MVC виступає у ролі контролеру. Воно містить логіку, як отримувати доступ до моделей та застосовувати відповідний шаблон.

3. Шаблони. Вони є формою представлення даних. Шаблони мають свою метамову та є одними з основних засобів виводу на екран.

4. URL – механізм зовнішнього доступу до представлень.

На рисунку 5.1 зображено MVT архітектуру Django.

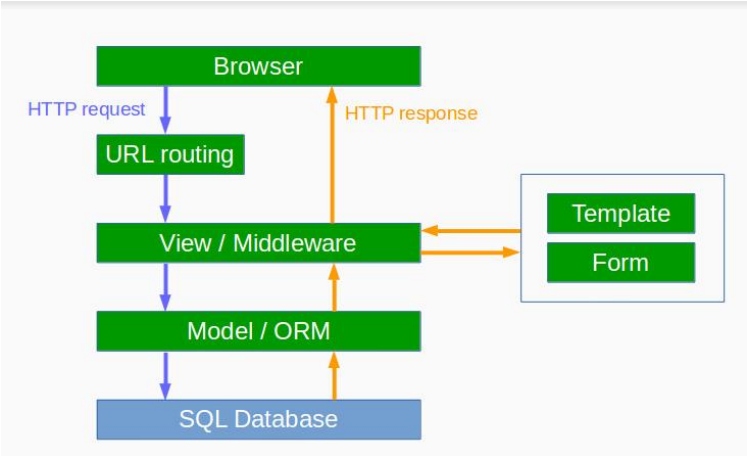

Рисунок 5.1 – Архітектура Django

Однією з особливостей фреймворку є те, що Django дозволяє генерувати HTML динамічно за допомогою шаблонів. Шаблони містять статичний HTML та динамічні дані, рендеринг яких описаний спеціальним синтаксисом.

Проект Django дозволяє використовувати один чи декілька механізмів створення шаблонів. Для цього реалізована система шаблонів, яка називається - мова шаблонів Django (Django template language, DTL). Ця система надає потужну і, водночас, просту мову розмітки, яка відповідає за надання даних користувачеві і сприяє розділенню рівня логіки та інтерфейсу.

Завдяки такій архітектурі, Django є легко масштабованим, а вбудовані засоби для спрощення процесу автентифікації користувача, роботи з картами сайту, адмініструванням вмісту тощо дозволяють пришвидшити кожен з етапів веб-розробки.

5.2. Функціонал додатку

Додаток має наступний функціонал: реєстрація, авторизація, завантаження відео, обробка відео, а саме розпізнавання людей на кадрі, їх ідентифікація та подальше відстеження.

Функціонал додатку розбито по сторінкам, що розроблені за допомогою мов HTML, CSS та JavaScript, а також django-templates. В якості веб-фреймворку використовується Django.

Програмний додаток містить у собі одного головного актора – користувача системи. Користувач може зареєструватись, увійти у систему, додати або видалити свого працівника, подивитись статистику про проведенню часу працівником в заданих кімнатах, загальну статистику по кімнатах та додавати відео, з яких ця інформація отримується.

На рисунку 5.2. представлена діаграма прецедентів, яка описує функції та дії актора у системі.

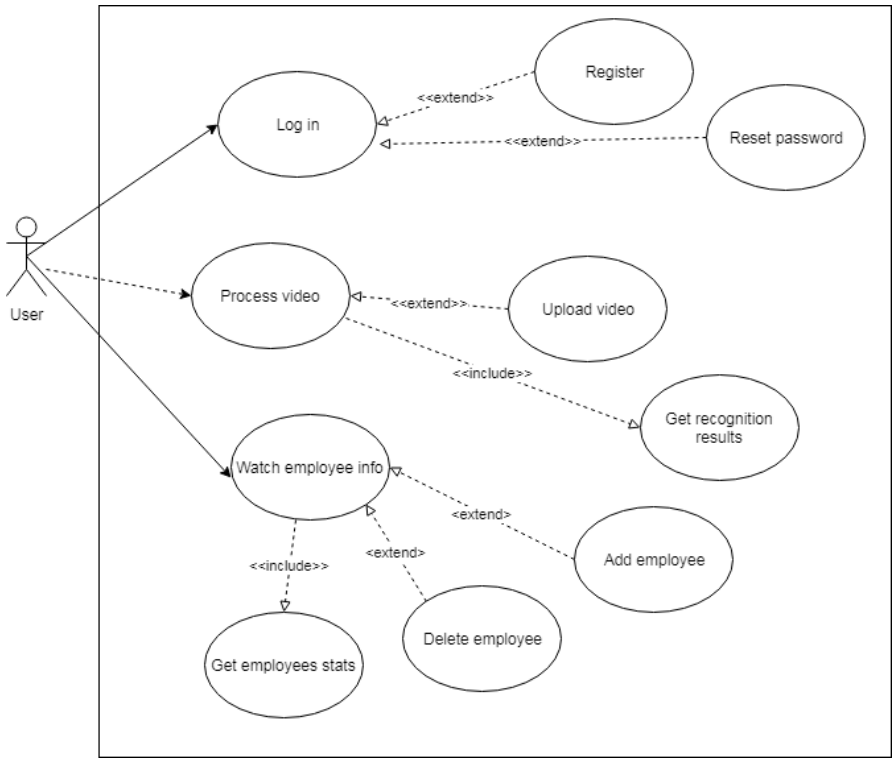

Рисунок 5.2 – Діаграма прецедентів

Процес знаходження людини виконується таким чином: спочатку використовується алгоритм для розпізнавання тіла та обличчя людини, він повертає координати боксів облич та тіла. Координати боксів облич далі передаються в алгоритмі для ідентифікації людини, який кодує дане зображення у вектор чисел та знаходить відповідність у базі даних. Далі, поки об’єкт не вийде за межі кадру, алгоритм трекінгу буде відстежувати його місцезнаходження.

На рисунку 5.3 схематично зображено принцип роботи алгоритму.

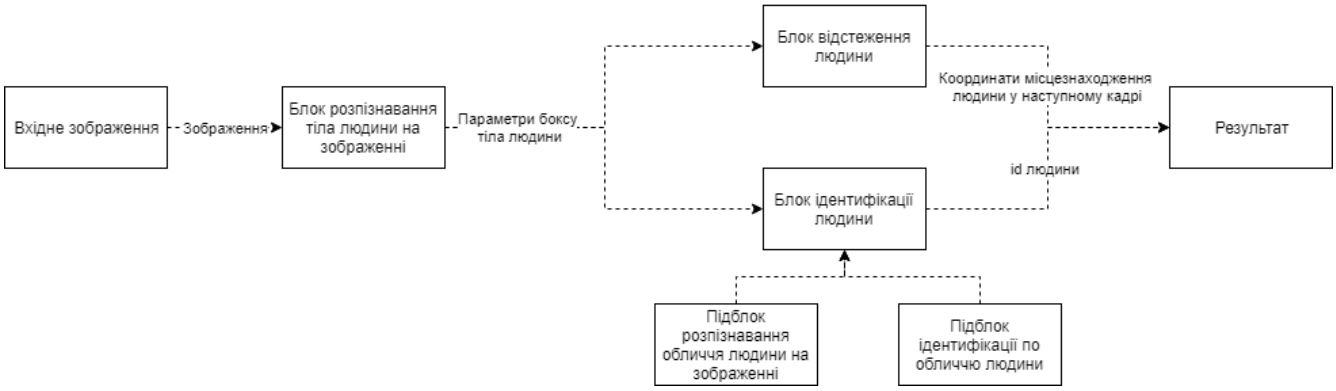

Рисунок 5.3 — Процес розпізнавання людини

5.3. Висновки до розділу

У даному розділі було описано структуру веб-додатку та основний інструмент, який було використано для його створення, а також наведено схему обробки відео.

6. ОПИС ВЗАЄМОДІЇ КОРИСТУВАЧА ІЗ СИСТЕМОЮ

У даному розділі будуть розглянуті основні етапи взаємодії користувача з веб системою.

6.1. Інтерфейс користувача

При вході на клієнтський додаток, користувачеві необхідно авторизуватися в системі щоб почати працювати в системі. На рисунку 5.1 зображена форма авторизації.

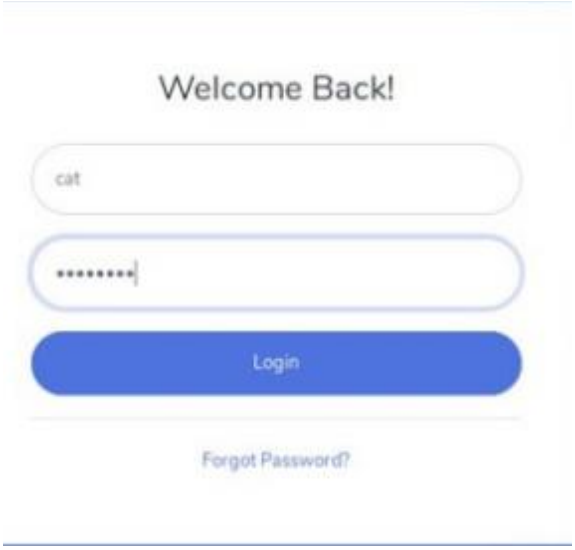

Рисунок 6.1 — Форма авторизації

Якщо користувач не може згадати пароль, йому необхідно вказати свою почтову адресу та отримати лист з новим паролем. На рисунку 5.2 зображена форма відновлення паролю.

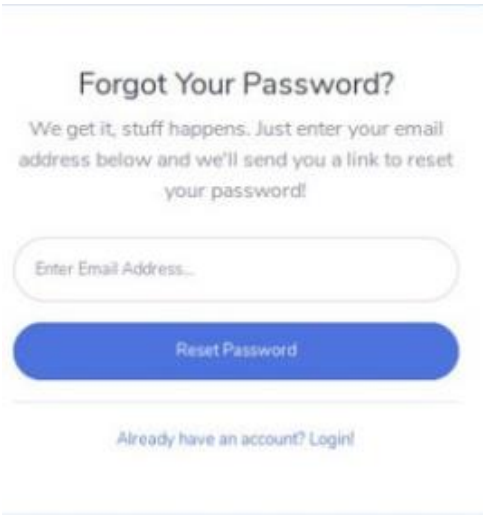

Рисунок 6.2 — Форма відновлення паролю

Після того, як користувач авторизувався в системі, він отримує доступ до головного меню системи. За допомогою цього меню користувач має доступ до усіх сторінок системи, або має можливість вийти з свого профілю. На рисунку 5.3 зображене головне меню системи.

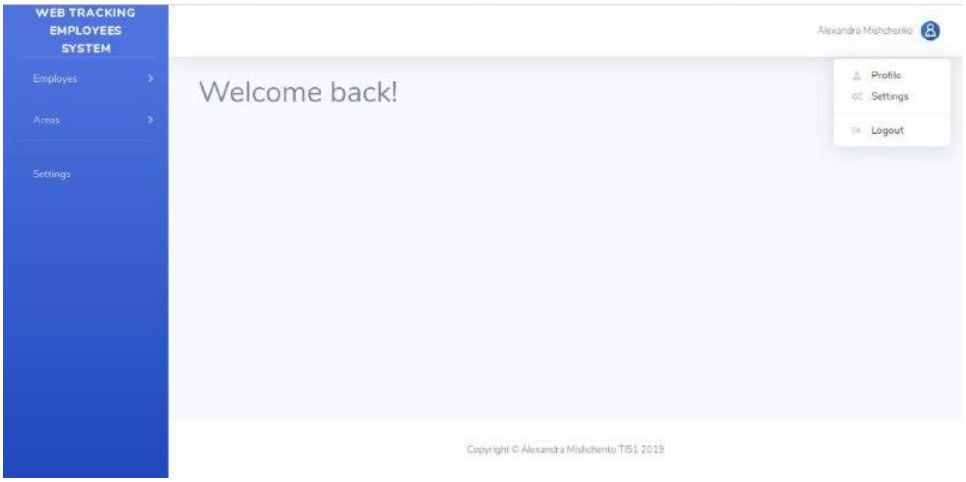

Рисунок 6.3 — Головне меню системи

Одним із головних компонентів кабінету користувача є меню керування кімнатами. За його допомогою користувач має змогу завантажувати відео для кожної з кімнат або підключати камеру відеоспостереження.

Користувач може завантажити відео та отримати по ньому результат. На рисунку 6.4 зображено процес обробки відео.

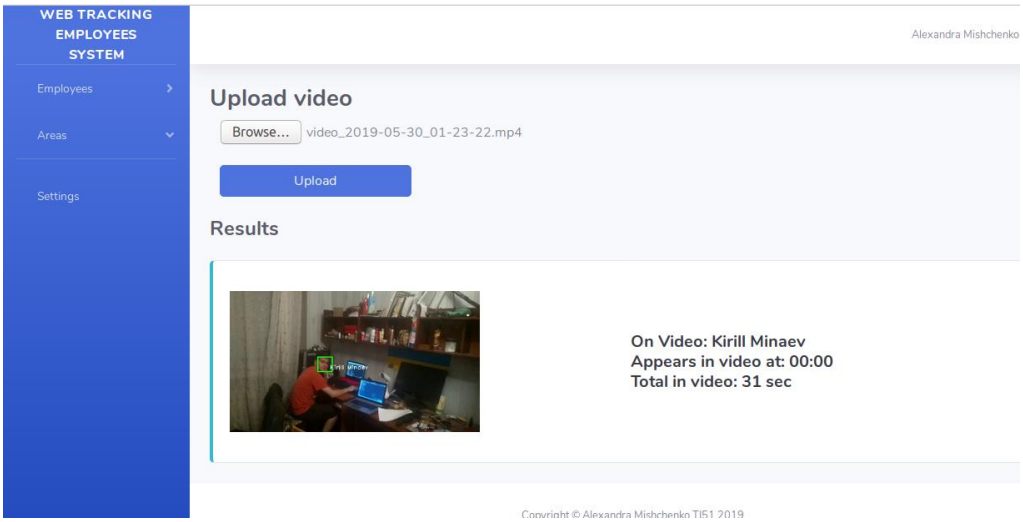

Рисунок 6.4 – Обробка відео

Після того, як відео опрацьовано, усі дані зберігаються та додаються до даних працівників.

У розділі працівників менеджер може переглянути статистику по кожному зі своїх працівників. Якщо у менеджера з'являється новий підопічний, він може обрати його зі списку усіх працівників компанії або, якщо його не встигли занести у базу, додати його самостійно.

Процес роботи з профілем працівника представлено на рисунках 6.5, 6.6, 6.7.

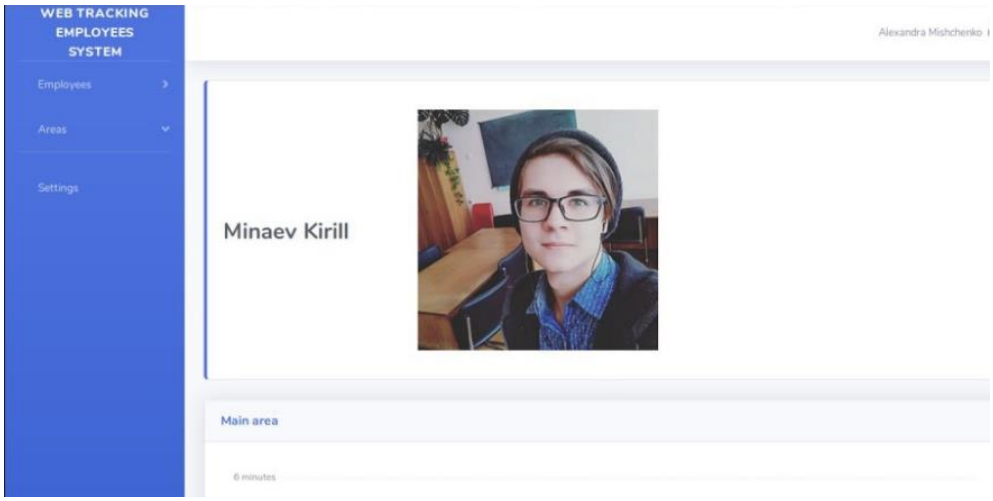

Рисунок 6.5 - Інформація про працівника

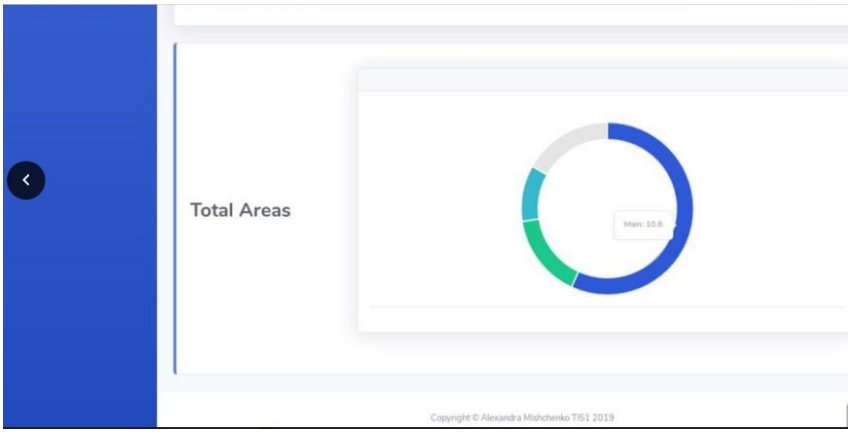

Рисунок 6.6 - Інформація про час, проведений працівником в усіх приміщеннях

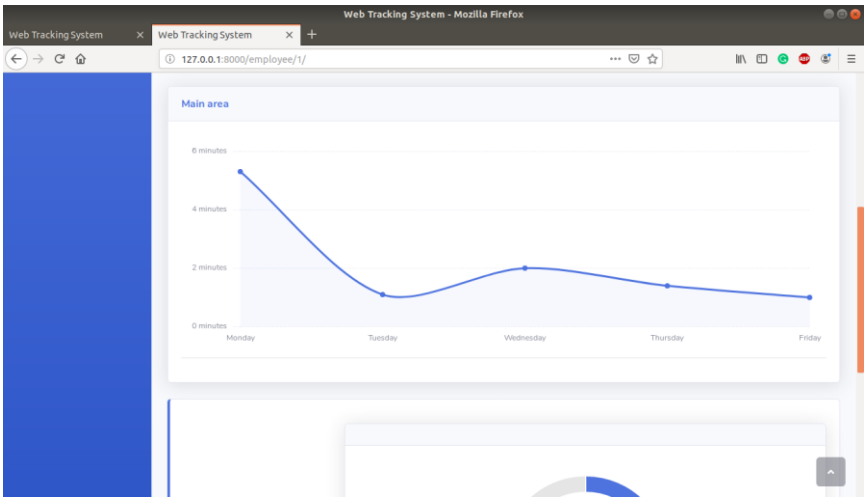

Рисунок 6.7 - Інформація про час, проведений працівником в головному приміщенні

6.2. Висновки до розділу

В процесі створення системи ідентифікації та трекінгу об’єктів було вирішено задачу знаходження та відстеження декількох людей на відео, а також ідентифікацію знайдених людей. Розроблено інтерфейс користувача, що з легкістю дозволяє завантажити керувати працівниками, переглядати інформацію про них, завантажувати файл відеозапису та переглядати опрацьовані дані.

ВИСНОВКИ

Під час роботи над дипломним проектом було проаналізовано проблему задач трекінгу та ідентифікації об’єктів на відео та розглянуто технології, за допомогою було побудовано веб-систему трекінгу та ідентифікації об’єктів у відеопотоці. Було досліджено три подібні системи, з чого було зроблено висновок, що ці системи не розв’язують повністю задану задачу. До того ж, було порівняно актуальні алгоритми для знаходження, відстеження та ідентифікації об’єктів та обґрунтовано вибір кожного використаного методу.

Як результат роботи було розроблено програмне забезпечення у вигляді веб системи по трекінгу та ідентифікації об’єктів у відеопотоці на прикладі задачі офісного менеджменту, а саме ідентифікації працівників у відеопотоці та відстеження їх переміщення та час проведення у заданому відео.

Для цього було використано мову програмування Python, веб-фреймворк Django, базу даних MongoDB та засоби бібліотек OpenCV та Dlib.

Кінцевий продукт реалізує усі задачі, поставлені у Розділі 1, а саме:

- автентифікація, реєстрація користувача та можливість відновити пароль в разі його втрати;
- можливість завантаження відео для подальшої обробки та ідентифікації людей на ньому;
- можливість створення зон, в які необхідно завантажувати відповідні відеофайли;
- можливість перегляду кожного з працівників користувача та створену статистику, яка показує, скільки часу було проведено в тій чи іншій зоні;
- зручний інтерфейс для використання системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. China verification security system[Електронний ресурс] – Режим доступу до ресурсу: https://clck.ru/GWCGt
2. SoloShot3 [Електронний ресурс] – Режим доступу до ресурсу: https://soloshot.com/collections/soloshot3
3. Vision4ce [Електронний ресурс] – Режим доступу до ресурсу: https://www.vision4ce.com/
4. AR Tracking System [Електронний ресурс] – Режим доступу до ресурсу: https://ar-tracking.com/products/tracking-systems/
5. Viola-Jones Overview [Електронний ресурс] – Режим доступу до ресурсу: https://clck.ru/GWCJd
6. Haar Cascade Detector using OpenCV [Електронний ресурс] – Режим доступу до ресурсу: https://api-2d3d-cad.com/viola-jones-method/
7. Ерік Бахан — CVPR 2019. Papers overview. [Електронний ресурс]. — 2018. — Режим доступу: https://syncedreview.com/2019/02/28/cvpr-2019-accepts-record-1300-papers/
8. Емілія Марк — Papers comprehension 2016-2019. [Електронний ресурс]. — 2018. — Режим доступу: https://medium.com/syncedreview/cvpr-2018-kicks-off-best-papers-announced-d3361bcc6984
9. Анжеліна Бегенс — Object Tracking with OpenCV. Papers overview. [Електронний ресурс]. — 2018. — Режим доступу: https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/
10. Марк Лутц — Python для професіоналів [Електронний ресурс]. — 2018. — Режим доступу: https://bit.ly/2JasdJFe.
11. Ерик Метиз — Розробка веб-додатків на Python [Електронний ресурс]. — 2018. — Режим доступу: https://bit.ly/2AsfgGD7.

12. Марку Ігер — MVC Guide for Enterprise Architectures in Python [Електронний ресурс]. — 2015. — Режим доступу: https://www.packtpub.com/application-development/mvc-survival-guide-enterprise-architectures-python.
13. Нигел Георг — Mastering Django: Core [Електронний ресурс]. — 2015. – Режим доступу: https://bit.ly/5slgCXtzR.
14. Деніел Рой — Two Scoops of Django: Best Practices [Електронний ресурс]. — 2015. — Режим доступу: https://bit.ly/sdf512e.
15. Болье А. — Learning NoSQL [Електронний ресурс]. — 2014. — Режим доступу: http://shop.oreilly.com/product/9780596007270.do.
16. Нейронные сети для начинающих [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/post/312450/
17. L. Deng. Deep Learning: Methods and Applications / L. Deng, D. Yu., 2014.

ДОДАТОК А

Web-система трекінгу та ідентифікації об’єктів у відеопотоці

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ51185_18Б

Аркушів 2

Київ 2019

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ51185_18Б	Записка.docx	Текстова частина дипломної роботи
Компоненти		
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ51185_18Б 12-1	recognizer.py	Основний компонент
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТІ51185_18Б 12-1	manage.py	Основний компонент

ДОДАТОК Б

Web-система трекінгу та ідентифікації об’єктів у відеопотоці

Текст програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ51185_18Б 12-1

Аркушів 6

Київ 2019

```
import face_recognition
import cv2
import glob
from pymongo import MongoClient
from bson import ObjectId
import os
from employees.models import Employee


class EmployeeHandler():
def __init__(self):
super(EmployeeHandler, self).__init__()
self.employees = Employee.objects.all()
self.fill_collection()

def get_collection(self):
return self.clients.find({})

def get_employee(self, _id):
return self.employees.find_one({'_id':ObjectId(_id)})

def get_ids(self):
return self.employees.find({}, { '_id':1})

def get_encodings(self):
return self.employees.find({}, { 'embedding':1, '_id':0})

def get_clients(self, ids):
return self.employees.find({ "_id": { "$in": ids }})

def isExist(self, name, surname):
return self.employees.find({"name": name, "surname": surname}).count() > 0

@staticmethod
def fill_collection():
for employee in Employee.objects.all():
if employee.embedding is None and os.path.exists(employee.photo.url):
image = face_recognition.load_image_file(employee.photo.url)
name, surname = employee.name, employee.surname
encoding = face_recognition.face_encodings(image)[0].tolist()
employee.embedding = encoding
employee.save()


def add_client(self, client):
try:
self.employees.insert_one(client)
except:
raise Exception('Could not add to database')
```

```
import face_recognition
import cv2
import sys
from employee.models import Employee

def recognize_on_frame(frame, employees):
'''
Recognize people on the frame comparing them with database
Args:
frame(nparray): current frame
'''
if frame is None:
raise Exception('You did not pass any image')
if self.transform:
frame = self.transform(frame)
face_ids = []
face_locations = face_recognition.face_locations(frame)
face_encodings = face_recognition.face_encodings(frame, face_locations)
users = []
for face_encoding in face_encodings:
# See if the face is a match for the known face(s)
encodings = [employee.embedding for employee in employees]
ids = [employee._id for employee in employees]

matches = face_recognition.compare_faces(encodings, face_encoding)

# If a match was found in known_face_encodings, just use the first one.
if True in matches:
first_match_index = matches.index(True)
_id = ids[first_match_index]
face_ids.append(_id)
users.append(Employee.objects.get(pk=this_object_id))
else:
users.append('Unknown')
return users, face_locations


def draw_results(frame, face_locations, employees):
'''
Draws bounding boxes with predicted names on the frame
Args:
frame(nparray): current frame
face_locations(list): locations of each face on the frame
face_ids(list): ids of each person founded on the frame
'''
for (top, right, bottom, left), client in zip(face_locations, list(users)):
Draw a box around the face
cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

Draw a label with a name below the face
```

```
cv2.rectangle(frame, (left, bottom - 10), (right, bottom), (0, 0, 255), cv2.FILLED)
font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(frame, self.db_manager.get_fullname(client), (left + 6, bottom - 6), font, 0.5, (255, 255, 255), 1)
return frame

def process_video(path):

cascPath = 'haars.xml'
faceCascade = cv2.CascadeClassifier(cascPath)

video_capture = cv2.VideoCapture(path)
tracker = multiTracker = cv2.MultiTracker_create()
ret, frame = video_capture.read()

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

faces = faceCascade.detectMultiScale(
gray,
scaleFactor=1.1,
minNeighbors=5,
minSize=(30, 30)
)

people = recognize_on_frame(frame, Employy.objects.all())
#for face in faces:
(x, y, w, h) = [int(v) for v in face]
tracker.add(cv2.TrackerGOTURN_create(), frame, [x, y, w,h])
old_boxes = faces
person = []
time = 0
while True:
Capture frame-by-frame
ret, frame = video_capture.read()

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

success, boxes = tracker.update(frame)
#Draw a rectangle around the faces
for (x, y, w, h) in boxes:
cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

if len(boxes) > len(old_boxes):
new_boxes = [box for box in boxes if box not in old_boxes]
for box in new_boxes:
tracker.add(cv2.TrackerGOTURN_create(), frame, box)
people.append(recognize_on_frame(frame, Employee.objects.all())

if len(boxes) < len(old_boxes):
gone_boxes = [box for box in old_boxes if box not in boxes]
```

```
gone_indexes = []
for i, box in enumerate(boxes):
for gone_box in gone_boxes:
if gone_box == box:
employee = Employee.get(people[i])
employee.end_time = time

font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(frame, person, (x + w - 6, y + h - 6), font, 0.5, (255, 255, 255), 1)
Display the resulting frame
time+=1
cv2.imshow('Video', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):
break

#When everything is done, release the capture
video_capture.release()
cv2.destroyAllWindows()



if __name__=="__main__":
process_video("video_2019-05-30_01-23-22.mp4")

from django.shortcuts import render
from django.views.generic import TemplateView, DetailView
from django.core.files.storage import FileSystemStorage

# Create your views here.

class VideoView(TemplateView):
template_name = 'videos/video_new.html'

def post(self, request):
if request.FILES:
myfile = request.FILES['myfile']
fs = FileSystemStorage()
filename = fs.save(myfile.name, myfile)
uploaded_file_url = fs.url(filename)
return render(request, 'videos/video_new.html', {'uploaded_file_url': uploaded_file_url})
return render(request, 'videos/video_new.html')


from django.shortcuts import render

from django.views.generic import ListView, DetailView
from django.views.generic.edit import CreateView

from .models import Employee
```

```
from .forms import NewEmployeeForm

class EmployeeListView(ListView):
model = Employee
template_name = 'home.html'

import json

class EmployeeDetailView(DetailView):
model = Employee
template_name = 'employees/employee_detail.html'

def get_context_data(self, *args, **kwargs):
context = super().get_context_data(*args, **kwargs)
context['data'] = employee.start_time, employee.end_time
print(context)
return context

class EmployeeCreateView(CreateView):
model = Employee
template_name = 'employees/register.html'
form_class = NewEmployeeForm

def post(self, request):
print(request.FILES)
employee = self.model()
employee.manager = request.user
employee.name = request.POST['name']
employee.surname = request.POST['surname']
employee.photo = request.FILES['photo']

employee.save()
print(employee)
#return HttpResponseRedirect('/')

return render(request, 'employees/employee_detail.html', {'object': employee})
```

ДОДАТОК В

Web-система трекінгу та ідентифікації об’єктів у відеопотоці

Опис програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ51185_18Б 13-1

Аркушів 10

Київ 2019

АНОТАЦІЯ

Додаток містить опис основного компоненту системи трекінгу та ідентифікації об’єктів та задач, поставлених в розділі 1, а саме:

- знаходження, відстеження та ідентифікацію людини у кадрі;
- відображення результатів у зрозумілому вигляді для користувача;
- розробка веб-додатку зі зручним інтерфейсом для користувача.

Додаток розроблений за допомогою мови програмування Python та фреймворку Django.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ

Даний додаток містить опис основного компоненту веб системи для задач трекінгу та ідентифікації об’єктів. У додатку Б міститься програмний код компоненту.

Система працює у вигляді веб-додатку. Для того, щоб її розвернути локально, необхідно встановити Python 3.6, фреймворк Django та базу даних MongoDB.

Додаток розроблений за допомогою мови програмування Python та фреймворку Django.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Веб система надає наступний функціонал користувачу:

- реєстрація користувачів;
- реєстрація користувачами працівників;
- автентифікація користувачів;
- завантаження відео на обробку;
- перегляд інформації про працівника;
- перегляд статистики про час знаходження працівника в головній та інших зонах.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Так як для побудови системи було використано фреймворк, який є реалізацією шаблону проектування MVC, а, якщо бути точнішим, MVT, відповідно й додаток складається з трьох логічних частин: шаблону, представлення та моделі.

Шаблони реалізовані на спеціальній мові Django-templates з залученням HTML, CSS та JavaScript та відповідають за відображення даних користувачу.

Представлення слугують для обробки HTTP запитів, здійснення викликів до сервісів розпізнавання та наповнення шаблонів даними.

Моделі призначені для здійснення запитів до бази даних та роботи з нею.

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для використання розробленого додатку користувач повинен мати персональний комп’ютер або мобільний пристрій з браузером, що підтримує JavaScript.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Програма не потребує інсталяції.

Для роботи з програмою достатньо відкрити браузер на сторінці додатку та обрати потрібний функціонал.

ВХІДНІ ДАНІ

Вхідна інформація для додатку:

1) дані для створення облікового запису;
2) фотографії працівників;
3) відео для обробки.

ВИХІДНІ ДАНІ

Вихідна інформація додатку:

1) інформація по обробленому відео;

2) статистика по кожному з працівник про його час проведений в заданих зонах.

ДОДАТОК Г

Web-система трекінгу та ідентифікації об’єктів у відеопотоці

Тези на конференцію “Сучасні проблеми наукового забезпечення енергетики”

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ511845_18Б

Аркушів 3

Київ 2019

Сучасний світ неможливо уявити без технологій. Кожного дня людина контактує з великою кількістю інформаційних систем, велику частку яких займає відеоспостереження. Магазини, зупинки, громадський транспорт, житлові будинки, вулиці в наш час забезпечені сучасними системами відеоспостереження, завдяки яким можна зробити життя людини безпечнішим та простішим: ідентифікувати небезпечних злочинців, знаходити зниклих людей, запобігати терактам. Водночас, такі системи можуть використовуватись для аналізу поведінки людей з метою покращення якості сервісу.

Існуючі алгоритми дозволяють знаходити, ідентифікувати та відстежувати об'єкти у відеопотоці. У задачі розпізнавання образів та знаходження об'єктів особливого успіху досягли згорткові нейронні мережі, які за допомогою згортки дозволяють ефективно знаходити ознаки, а завдяки багатошаровій архітектурі знаходити комплексні особливості серед великих обсягів даних. Сіамські нейроні мережі, в свою чергу, широко використовуються для ідентифікації об'єктів[1]. Найбільша перевага сіамських нейронних мереж полягає у тому, що вони дозволяють розрізняти об'єкти одного класу по єдиному зображенню (наприклад, для ідентифікації окремої людини в натовпі достатньо фото обличчя цієї людини).

Альтернативою згортковим мережам для ідентифікації об’єктів на зображенні є класичні алгоритми галузі комп'ютерного зору (наприкад, SIFT[2]), проте вони дуже програють нейронним мережам у точності.

Тим не менш, серед алгоритмів трекінгу об'єктів перевага надається класичним алгоритмам (Predator[3], Particle Filter[4]), так як вони потребують менших обчислювальних ресурсів, що значно прискорює швидкість роботи в режимі реального часу.

Метою роботи є створення системи ідентифікації та трекінгу об'єктів у відеопотоці та збереження отриманих результатів у базу даних для можливості подальшого аналізу та використання.

Базуючись на проведених дослідженнях, зроблено висновок про актуальність розробки власного програмно-апаратного комплексу для ідентифікації та трекінгу об'єктів у відеопотоці.

Перелік посилань:

1. Gregory Koch, Richard Zemel, Ruslan Salakhutdinov. Siamese Neural Networks for One-shot Image Recognition [Електронний ресурс]. – Режим доступу : https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

2. David G. Lowe. Object Recognition from Local Scale-Invariant Features [Електронний ресурс]. – Режим доступу : http://people.cs.ubc.ca/~lowe/papers/iccv99.pdf

3. Zdenek Kalal. OpenTLD a.k.a. Predator [Електронний ресурс]. – Режим доступу : http://kahlan.eps.surrey.ac.uk/featurespace/tld/

4. Arnaud Doucet, Adam M. Johansen. A Tutorial on Particle Filtering and Smoothing: Fifteen years later [Електронний ресурс]. – Режим доступу : https://www.seas.harvard.edu/courses/cs281/papers/doucet-johansen.pdf